qualified and have different lengths, the user must look both vertically (e.g., down) and then horizontally either to the left or right to view the simple name of the next fully qualified resource identifier. This can become quite tiresome for a user of a conventional graphical user interface.

In addition, when a window in a conventional graphical user interface displays the fully qualified resource identifier for a resource, but that window is not large enough in the horizontal direction to completely display the entire alphanumeric string for the fully qualified resource identifier, the simple name of the resource might not be visible to the user of the graphical user interface. The user might have to operate the horizontal scroll bar in order to scroll the display to the left so that the simple name of the resource is visually apparent.

Since software applications must generally reference resources in a fully qualified manner, users must supply resource identifiers in a fully qualified format which requires that user to perform error-free entry (e.g., typing) of the complete hierarchy location such as a full path name followed by the correct simple name of a particular resource. This can become quite cumbersome when referencing many resources or only a single resource having a long hierarchy location.

If a user decides to use the alias or shortcut feature of a conventional operating system to provide an second identification of a resource in a location other than the true location of that resource with the hierarchy (e.g., with the file system), the alias or shortcut for that resource might not provide an indication of the true location of that resource which it references. In other words, if the user creates a shortcut and gives that shortcut a make like "myfile," the graphical user interface will not automatically indcate the true location of the file that corresponds to the myfile shortcut. To determine this information, the user might have to select the alias or shortcut and activate (e.g., via a right mouse click) a pull-down menu to select a "Properties" feature in order to determine the actual file system location of the resource referenced by the alias or shortcut. If multiple shortcuts have the same simple name, it may be unclear to the user which shortcut refers to which resource. In other words, by viewing the simple name of the alias or shortcut by itself, the user will not automatically be provided with true identity or

location of the resource to which that alias or shortcut relates. If the user wants an alias to indicate the true identity of the resource to which that alias refers, then the user must manually name the alias or shortcut with an indication of the hierarchy location or resource identity. Since this process is not automated, inconsistencies in naming can result from one alias or shortcut to another and no warning is provided if the name the user selects is already in use by another resource.

In addition, conventional software applications and operating systems do not provide a sufficient mechanism to create a grouping or category for resources within a resource hierarchy such as a file system without having that grouping or category be included in the resource hierarchy location. For example, a user of a conventional operating system may create a sub-directory to hold files of a certain type. However, to reference those files, the user must supply that directory name as part of the path to the files. There is no technique in a conventional graphical user interface-based operating system to create a grouping directory for placement of files and then to reference those files without having to reference the grouping. In addition, directories or other grouping constructs provided by conventional software applications do not insulate resources located below that directory from user operations. For example, a user cannot place files into a directory, and then remove that directory, without removing the files. This is because conventional grouping techniques, such as creating directories to group files or other resources, force the directory to become part of the hierarchy location or pathname of the file or resource. There are no easy ways to create a grouping structure such as a directory within a conventional operating system but to have resource placed in that directory to be considered part of another directory instead.

These and other deficiencies common to conventional software and operating system naming schemes and resource referencing techniques can result in user errors since a user may perform an operation on a resource (e.g., deleting a file) without being fully aware of the identity of the resource. The user may thus accidentally reference an incorrect resource (e.g., as a result of a typographical error when entering a long resource identifier). To avoid such errors, users must continually be thinking about the naming scheme and hierarchical structure imposed upon resource identifiers so as to avoid

accidentally referencing the wrong resource. In addition, if resource identifiers are lengthy, user must provide significantly more manual graphical user interface operations (e.g., scrolling to identify simple names) to properly reference resources.

Conversely, embodiments of the present invention provide unique resource identification, naming, grouping and referencing techniques that an operating system and/or a software application using a graphical user interface can employ to significantly overcome many of the problems of conventional graphical user interfaces used for management of resources in a computer system, data storage, or computer network environment. Preferred embodiments of the invention operate within a management station computer system such as a storage area network management station. Such a computer system can operate, for example, a resource management application that provides the graphical user interface and resource representation techniques and mechanisms as explained herein.

In particular, the system of the invention provides method embodiments which operate in a computer system having a memory system and a display that displays a graphical user interface for management of network resources. The method embodiments operate to represent one or more resources in a computing system environment. One such method embodiment comprises the steps of creating an object to represent a resource in the computing system environment. A user of the computer system may instruct the management software to create the object, or alternatively, the management station may be configured with software that can "discover" resources that are capable of being managed and can create objects for each discovered resource. The details of the process of discovering resources that can be managed, for example, within components that exist within a storage area network environment is the subject of a co-pending patent application entitled "______________________________" having US Serial No. 09/__________, filed _____________ and which is assigned to the Assignee of the present invention. The object that is created according to this method is generally an instantiation of a data structure, such as an instantiation of a Java or C++ class, that contains data definitions and methods that describe the resource that the object represents.